Eventually, Understanding

Operationalizing Misconceptions in LTL

Siddhartha Prasad
Ben Greenman, Skyler Austen,
Tim Nelson, Kathi Fisler
Shriram Krishnamurthi
Brown University

Linear Temporal Logic (LTL)

Propositional logic (variables, *and*, *or*, *not*)

Temporal operators:

always(E)
eventually(E)
next_state(E)
E_1 until E_2

G (E) □(E)
F (E) ◇(E)
X (E) ○(E)
E_1 U E_2

eventually(green-traffic-light)
lock-acquired ➔ eventually(lock-released)
not(cookie) until homework-done

x U y

Option 1

x is true until time t;
at t,
y becomes true

Option 2

x is true until time t;
at t,
y becomes true
and x becomes false

x U y

Option 1

x is true until time t;
at t,
y becomes true

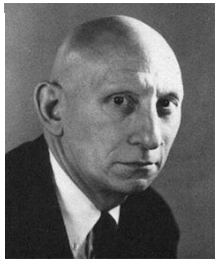
The robot won't leave the dock
until
its battery is full

Option 2

x is true until time t;
at t,
y becomes true
and x becomes false

The robot's motor is always on
until
it reaches the destination

One can't proceed from the informal to the formal by formal means.
- Alan Perlis



x U y

x is true until time t;
at t,
y becomes true
and x becomes false

Misconception!
Incorrect semantics in mental model

The robot's motor is always on
until
it reaches the destination

Identifying LTL misconceptions

Four rounds of study over 3 years

Over 150 students
29 researchers with prior LTL expertise

Multiple LTL syntaxes

Multiple question types
- Translate English to LTL
- Determine Trace Satisfaction
- Choose satisfying trace

Little Tricky Logic: Misconceptions in the Understanding of LTL

Ben Greenman[a], Sam Saarinen[a], Tim Nelson[a], and Shriram Krishnamurthi[a]

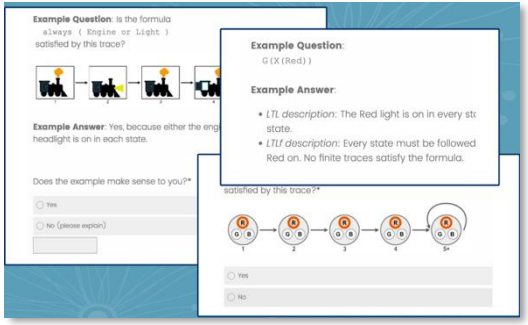
Misconceptions in Finite-Trace and Infinite-Trace Linear Temporal Logic

FM Artifact Evaluation Available

FM Artifact Evaluation Reusable

Ben Greenman(✉)[1,2][0000−0001−7078−9287], Siddhartha Prasad[2][0000−0001−7936−8147], Antonio Di Stasio[3][0000−0001−5475−2978], Shufang Zhu[3][0000−0002−5922−8750], Giuseppe De Giacomo[3][0000−0001−9680−7658], Shriram Krishnamurthi[2][0000−0001−5184−1975], Marco Montali[4][0000−0002−8021−3430], Tim Nelson[2][0000−0002−9377−9943], and Milda Zizyte[2][0000−0002−4823−8937]

[1] University of Utah, USA
[2] Brown University, USA
[3] University of Oxford, United Kingdom
[4] Free University of Bozen–Bolzano, Italy



1. **Precedence** (Eng ▷ LTL only): Applies to LTL formulas that are correct up to missing parentheses. For other misparenthesized formulas, apply no labels (Unlabeled) instead of guessing about intent.
   - Expected (x and y) => z but subject wrote x and y => z
2. **ReasonableVariant** (Eng ▷ LTL only): Applies to LTL formulas that are correct for an unintended reading of the question. The example here is based on the phrase "Blue will turn on." We intended for Blue to be on now or later, but some subjects (quite reasonably!) specified a flip from off to on. Rajhans et al. [67] call this a *rising edge* issue.
   - Expected F(Blue) but subject wrote (not Blue) and F(Blue)
3. **Unlabeled**: If an answer is convoluted and/or ambiguous, then apply this tag and no others.
4. **BadProp**: Applies to responses that mis-use a logical operator or an atomic symbol.
   - Expected "if x then y" but subject wrote "both x and y hold"
   - Expected G(x) but subject wrote G(y)
5. **BadStateIndex**: Applies to responses that use a correct term at an incorrect apply when a fan-out operator (F, G, U) is missing or included erroneously.
   - Expected "x holds three states from now" but subject wrote "x holds now"
   - Expected x U (y and F(z)) but subject wrote (x U y) and F(z)
6. **BadStateQuantification**: Applies to responses that mis-use or swap a fan-out
   - Expected "x eventually holds" but subject wrote "x always holds"
   - Expected x U y but subject wrote G(x) U y
7. **ExclusiveU**: Applies to responses that assume an until is satisfied only when both the right subterm and the negation of the left subterm hold.
   - Expected x U ((not x) and y) but subject wrote x U y
8. **ImplicitF**: Applies to responses that either ignore or introduce an F quantifier.
   - Expected "whenever x, then y in the next state" but subject wrote "x and y alternate forever"
   - Expected F(x) but subject wrote x
9. **ImplicitG**: Applies to responses that either ignore or introduce a G quantifier.
   - Expected "x always holds" but subject wrote "x holds now"
   - Expected G(x => G(y)) but subject wrote G(x => y)
10. **OtherImplicit**: Applies to underconstrained responses that are not covered by the ImplicitF and ImplicitG tags.
    - Expected "whenever x holds then y also holds" but subject wrote "x holds now and y holds"
    - Expected (not x) U x but subject wrote F(x)
    - Expected x U G(not x) but subject wrote x and F(G(not x))
    - Expected F(x) and G(x => X(G(not x))) but subject wrote F(x and X(G(not x)))
11. **WeakU**: Applies to responses that confuse the U operator with the weak variant W, which does not guarantee that its second subterm eventually holds.
    - Expected "x holds for a finite number of states" but subject wrote "x always holds"
    - Expected x U y but subject wrote F(y) and x U y

Little Tricky Logic: Misconceptions in the Understanding of LTL

Ben Greenman[a], Sam Saarinen[a], Tim Nelson[a], and Shriram Krishnamurthi[a]

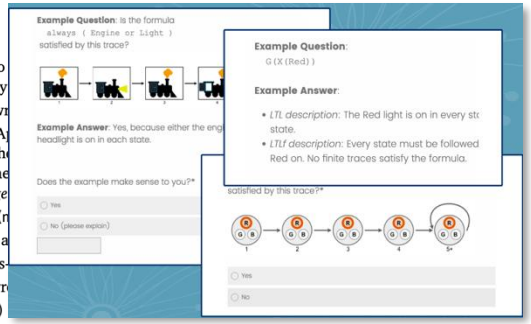
Misconceptions in Finite-Trace and Infinite-Trace Linear Temporal Logic

FM Artifact Evaluation Available

FM Artifact Evaluation Reusable

Ben Greenman(✉)[1,2][0000−0001−7078−9287], Siddhartha Prasad[2][0000−0001−7936−8147], Antonio Di Stasio[3][0000−0001−5475−2978], Shufang Zhu[3][0000−0002−5922−8750], Giuseppe De Giacomo[3][0000−0001−9680−7658], Shriram Krishnamurthi[2][0000−0001−5184−1975], Marco Montali[4][0000−0002−8021−3430], Tim Nelson[2][0000−0002−9377−9943], and Milda Zizyte[2][0000−0002−4823−8937]

[1] University of Utah, USA
[2] Brown University, USA
[3] University of Oxford, United Kingdom
[4] Free University of Bozen–Bolzano, Italy



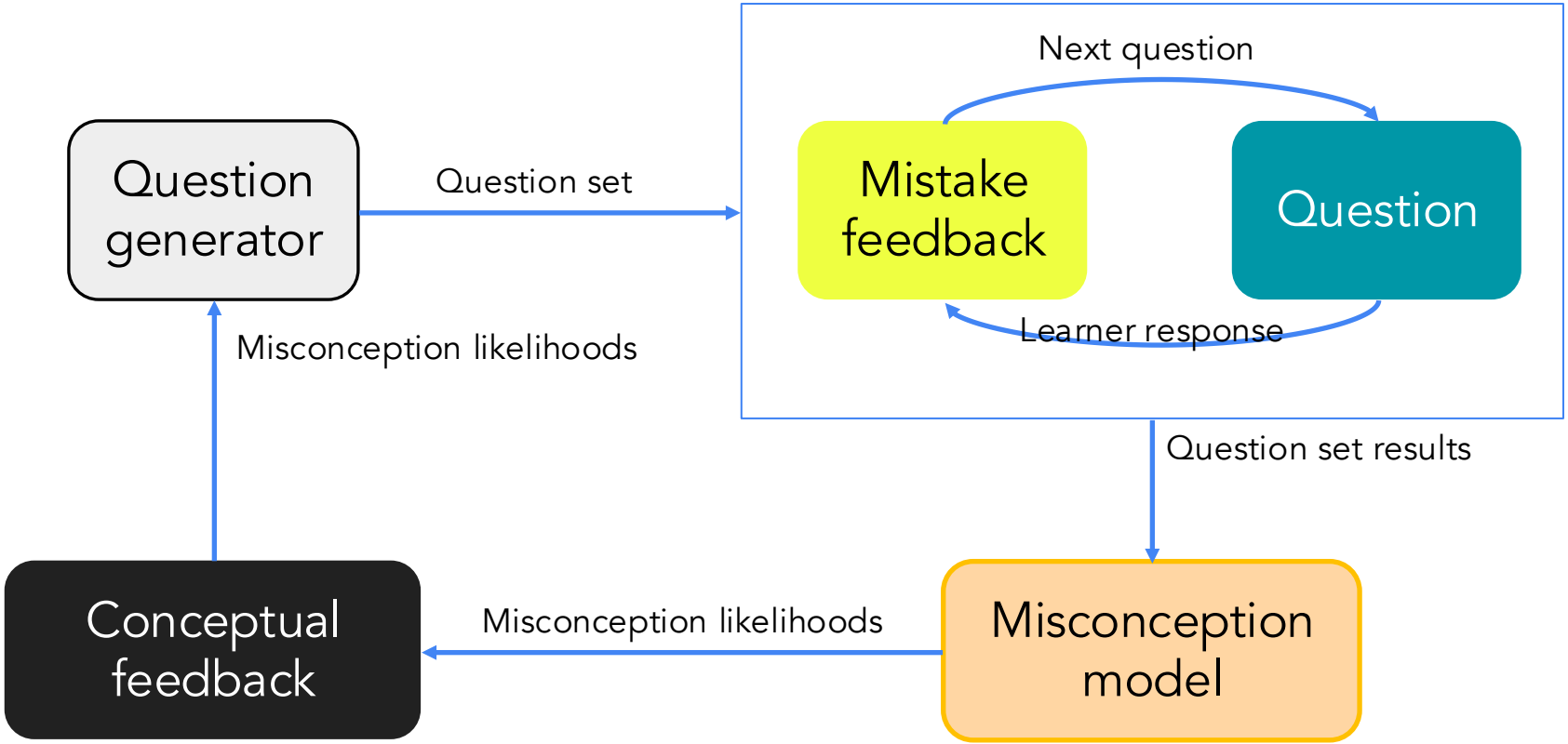
1. **Precedence** (Eng▷LTL only): Applies to other misparenthesized formulas, apply
   - Expected (x and y) => z but subject wr
2. **ReasonableVariant** (Eng▷LTL only): A reading of the question. The example h for Blue to be on now or later, but some Rajhans et al. [67] call this a *rising edge*
   - Expected F(Blue) but subject wrote (r
3. **Unlabeled**: If an answer is convoluted a
4. **BadProp**: Applies to responses that mis-
   - Expected "if x then y" but subject wr
   - Expected G(x) but subject wrote G(y)
5. **BadStateIndex**: Applies to responses that use a correct term at an incorrect state index. Does not apply when a fan-out operator (F, G, U) is missing or included erroneously.
   - Expected "x holds three states from now" but subject wrote "x holds now"
   - Expected x U (y and F(z)) but subject wrote (x U y) and F(z)
6. **BadStateQuantification**: Applies to responses that mis-use or swap a fan-out operator (F, G, U).
   - Expected "x eventually holds" but subject wrote "x always holds"
   - Expected x U y but subject wrote G(x) U y

Alloy 6

Electrum

Better tools, logics, teaching

LTL Tutor


Next question
Question generator
Question set
Mistake feedback
Question
Learner response
Misconception likelihoods
Question set results
Conceptual feedback
Misconception likelihoods
Misconception model

Classical Mutation

The robot's motor is always on until it reaches the destination

$G(on\ U\ (\neg on \wedge dest))$

$G(on\ U\ \neg on) \wedge dest$

$G(on\ U\ (\neg on \vee dest))$

$G((\neg on \wedge dest))\ U\ on)$

Semantic Mutation

Mutate correct answers to embody misconceptions.

ImplicitG: Applies to responses that either ignore or introduce a G quantifier.

- Expected "x always holds" but subject wrote "x holds now"
- Expected G(x => G(y)) but subject wrote G(x => y)

The robot's motor is always on until it reaches the destination

$$G\alpha \xrightarrow{\text{mutate}} \alpha$$

G(on U (¬on ∧ dest))

on U (¬on ∧ dest)

Semantic Mutation

Mutate correct answers to embody misconceptions.

ExclusiveU: Applies to responses that assume an until is satisfied only when both the right subterm and the negation of the left subterm hold.

- Expected x U ((not x) and y) but subject wrote x U y

The robot's motor is always on until it reaches the destination

$$\alpha \boldsymbol{U} (\neg\alpha \wedge \beta) \xrightarrow{\text{mutate}} \alpha \boldsymbol{U} \beta$$

G(on U (¬on ∧ dest))

G(on U dest)

Semantic Mutation

Each misconception can be introduced via a principled set of rewrites

Misconception	Mutation Rules
Implicit $\boldsymbol{G}$	$\boldsymbol{G}\alpha \xrightarrow{\text{mutate}} \alpha$
Implicit $\boldsymbol{F}$	$\boldsymbol{F}\alpha \xrightarrow{\text{mutate}} \alpha$
Bad State	$\boldsymbol{G}\alpha \xrightarrow{\text{mutate}} \boldsymbol{F}\alpha$
Quantification	$\boldsymbol{F}\alpha \xrightarrow{\text{mutate}} \boldsymbol{G}\alpha$
	$\alpha \boldsymbol{U} \beta \xrightarrow{\text{mutate}} (\boldsymbol{F}\alpha)\boldsymbol{U}\beta$
	$\alpha \boldsymbol{U} \beta \xrightarrow{\text{mutate}} \alpha \boldsymbol{U}(\boldsymbol{F}\beta)$
	$\alpha \boldsymbol{U} \beta \xrightarrow{\text{mutate}} (\boldsymbol{G}\alpha)\boldsymbol{U}\beta$
	$\alpha \boldsymbol{U} \beta \xrightarrow{\text{mutate}} \alpha \boldsymbol{U}(\boldsymbol{G}\beta)$
	$\alpha \boldsymbol{U} \beta \xrightarrow{\text{mutate}} (\beta \boldsymbol{U} \alpha)$
Precedence	$\alpha \bowtie (\beta \bowtie \delta) \xrightarrow{\text{mutate}} (\alpha \bowtie \beta) \bowtie \delta$
Exclusive $\boldsymbol{U}$	$\alpha \boldsymbol{U}(\neg\alpha \wedge \beta) \xrightarrow{\text{mutate}} \alpha \boldsymbol{U} \beta$
Weak $\boldsymbol{U}$	$\alpha \boldsymbol{U} \beta \xrightarrow{\text{mutate}} (\alpha \boldsymbol{U} \beta) \wedge \boldsymbol{F}\beta$

Misconception	Mutation Rules
Bad State	$\delta \boldsymbol{U}(\alpha \wedge (\boldsymbol{F}\beta)) \xrightarrow{\text{mutate}} (\delta \boldsymbol{U}\alpha) \wedge (\boldsymbol{F}\beta)$
Index	$\delta \boldsymbol{U}(\alpha \wedge (\boldsymbol{G}\beta)) \xrightarrow{\text{mutate}} (\delta \boldsymbol{U}\alpha) \wedge (\boldsymbol{G}\beta)$
	$\delta \boldsymbol{U}(\alpha \wedge (\boldsymbol{F}\beta)) \xrightarrow{\text{mutate}} (\delta \boldsymbol{U}\alpha) \wedge (\boldsymbol{F}\beta)$
	$\delta \boldsymbol{U}(\alpha \vee (\boldsymbol{G}\beta)) \xrightarrow{\text{mutate}} (\delta \boldsymbol{U}\alpha) \vee (\boldsymbol{G}\beta)$
	$\delta \boldsymbol{U}(\alpha \implies (\boldsymbol{F}\beta)) \xrightarrow{\text{mutate}} (\delta \boldsymbol{U}\alpha) \implies (\boldsymbol{F}\beta)$
	$\delta \boldsymbol{U}(\alpha \implies (\boldsymbol{G}\beta)) \xrightarrow{\text{mutate}} (\delta \boldsymbol{U}\alpha) \implies (\boldsymbol{G}\beta)$
	$\boldsymbol{X}(\alpha \wedge \beta) \xrightarrow{\text{mutate}} (\boldsymbol{X}\alpha) \wedge \beta$
	$\boldsymbol{X}(\boldsymbol{X}(\boldsymbol{X} \ldots \boldsymbol{X}\alpha)) \xrightarrow{\text{mutate}} \boldsymbol{X}\alpha$
Other Implicit	$(\neg\alpha \boldsymbol{U} \alpha) \xrightarrow{\text{mutate}} \boldsymbol{F}\alpha$
	$(\alpha \boldsymbol{U}(\boldsymbol{G}\alpha)) \xrightarrow{\text{mutate}} \alpha \wedge (\boldsymbol{F}(\boldsymbol{G}\alpha))$
	$(\boldsymbol{F}\alpha) \wedge (\boldsymbol{G}(\beta \implies (\boldsymbol{X}(\boldsymbol{G}\beta)))) \xrightarrow{\text{mutate}} \alpha \wedge (\boldsymbol{X}(\boldsymbol{G}\beta))$
	$\boldsymbol{X}\alpha \xrightarrow{\text{mutate}} \boldsymbol{F}\alpha$
	$\alpha \bowtie \beta \xrightarrow{\text{mutate}} \alpha$
	$\alpha \bowtie \beta \xrightarrow{\text{mutate}} \beta$
	$\neg\alpha \xrightarrow{\text{mutate}} \alpha$

Are semantic mutants better distractors than syntactic mutants?

How can we help students learn from their mistakes?

Which of the following LTL formulae best represents this English sentence?

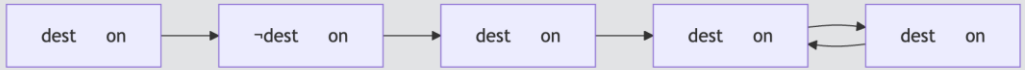
The robot's motor is always on until it reaches the destination

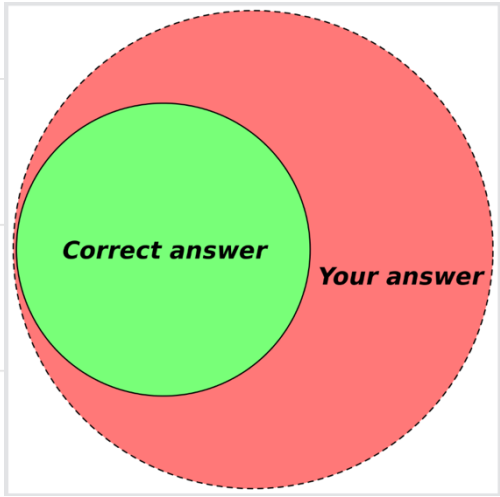
Is this English sentence confusing or unclear?

- ○ `(F (on U ((! on) & dest)))`
- ○ `G (on U (!on & dest))`
- ○ `(G ((on U ((! on) & dest)) | (G on)))`
- ○ `on U (!on & dest)`
- ○ `(G (on U dest))`

That's not correct 🙁 Don't worry, keep trying! The correct answer is highlighted in green (i.e: `G (on U (!on && dest))`)

Your selection is more permissive than the correct answer. Here is a trace that satisfies your selection, but not the correct answer:



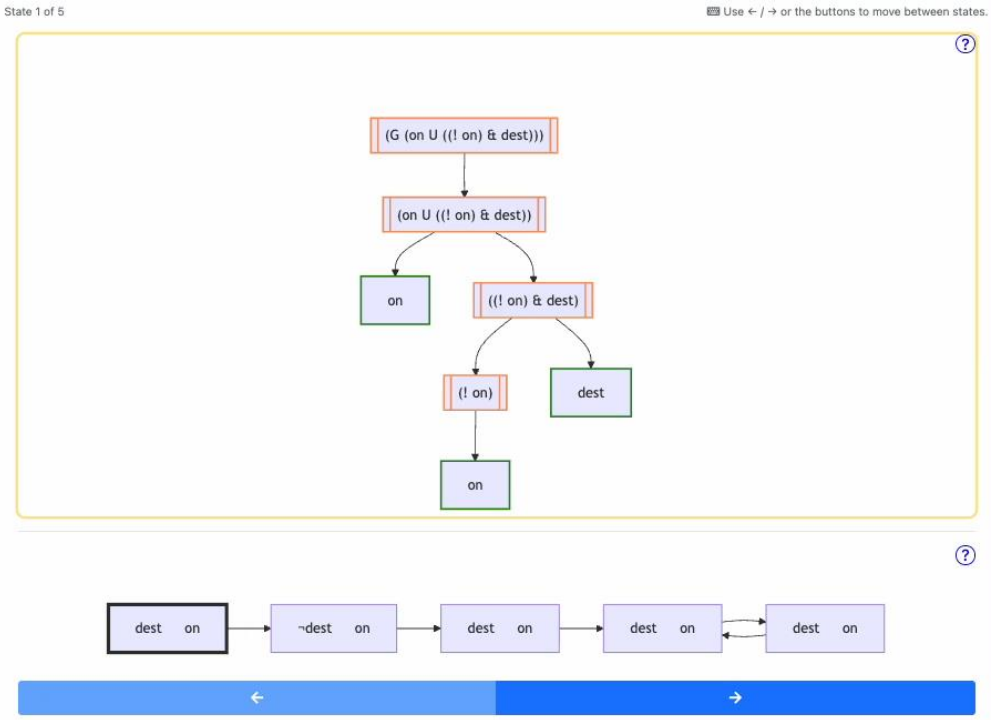


Which of the following LTL formulae best represents this English sentence?

The robot's motor is always on until it reaches the destination

Is this English sentence confusing or unclear?

- `(F (on U ((! on) & dest)))`
- `G (on U (!on & dest))`
- `(G ((on U ((! on) & dest)) | (G on)))`
- `on U (!on & dest)`
- `(G (on U dest))`



Which of the following LTL formulae best represents this English sentence?

The robot's motor is always on until it reaches the destination

Is this English sentence confusing or unclear?

- `(F (on U ((! on) & dest)))`
- `G (on U (!on & dest))`
- `(G ((on U ((! on) & dest)) | (G on)))`
- `on U (!on & dest)`
- `(G (on U dest))`

Learner may have the **Exclusive U** misconception

Until

Review

You might think that x U y (Until) is satisfied only when both the right subterm y and the negation of the left subterm x hold true. However, the "Until" operator is more permissive than this. In fact, it only requires that x holds true at every step until y becomes true. Once y holds, it doesn't matter whether x is true or false.

Until

Review

You might think that x U y (Until) is satisfied only when both the right subterm y and the negation of the left subterm x hold true. However, the "Until" operator is more permissive than this. In fact, it only requires that x holds true at every step until y becomes true. Once y holds, it doesn't matter whether x is true or false.

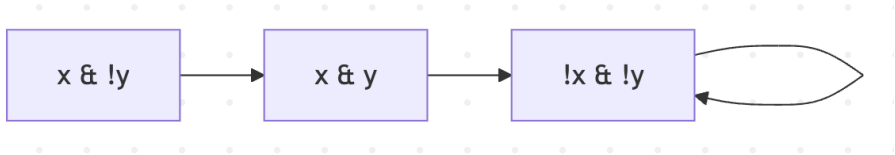
Until

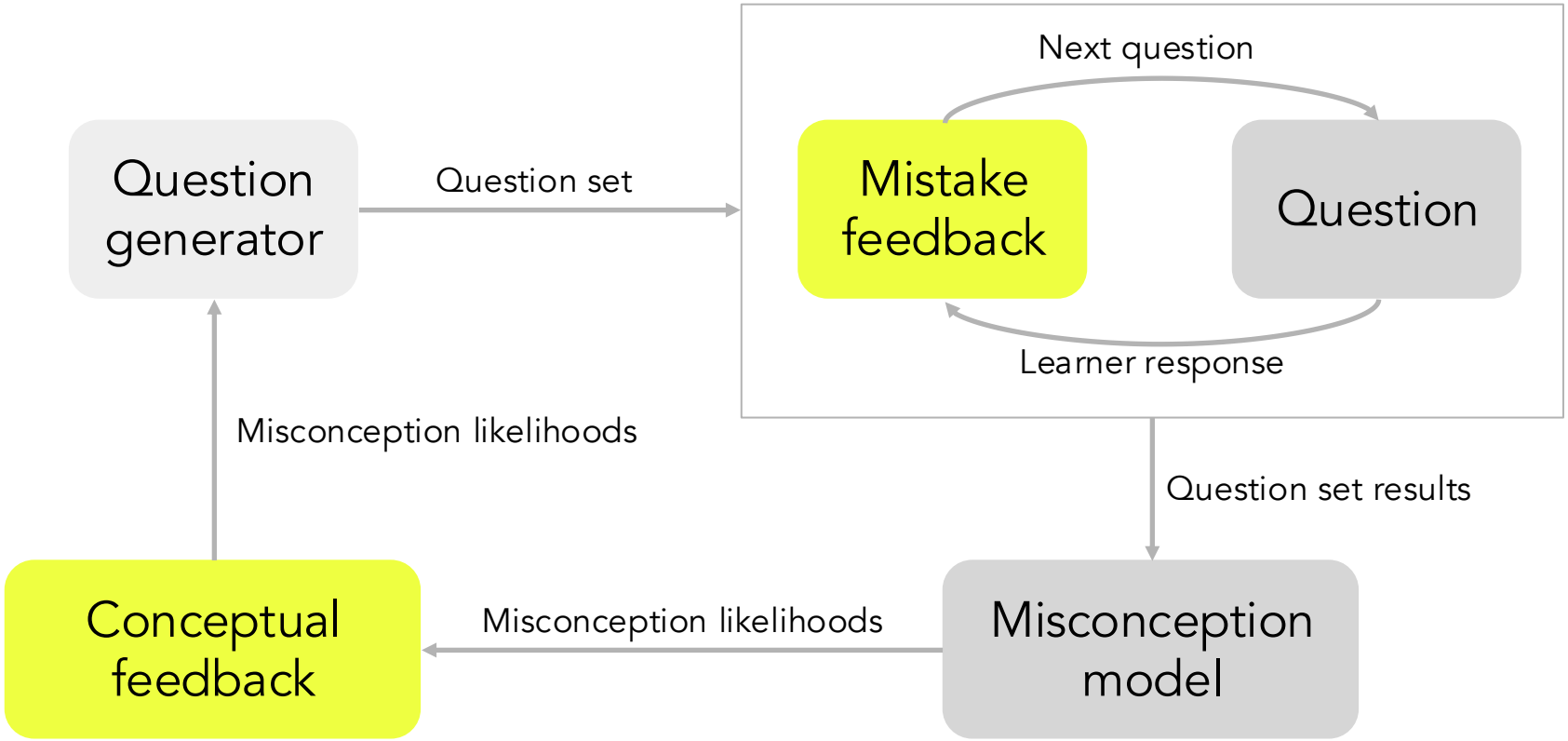
Review

You might think that x U y (Until) is satisfied only when both the right subterm y and the negation of the left subterm x hold true. However, the "Until" operator is more permissive than this. In fact, it only requires that x holds true at every step until y becomes true. Once y holds, it doesn't matter whether x is true or false.

To put it another way: y eventually becomes true at some point in the future, and *until* that moment, x must continuously hold true. This does not require x to stop being true at the same time that y becomes true.

For example, the following trace satisfies x U y :



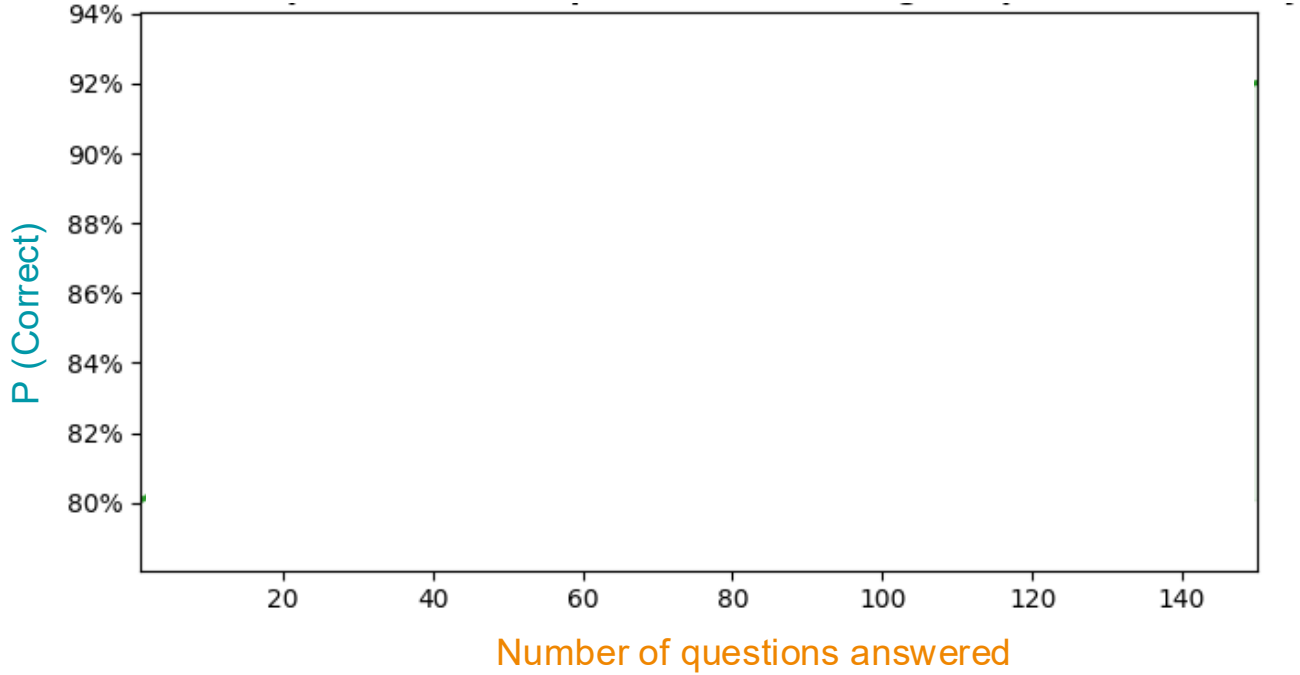

Next question
Question generator
Question set
Mistake feedback
Question
Learner response
Misconception likelihoods
Question set results
Conceptual feedback
Misconception likelihoods
Misconception model

Do learners using the tutor improve?

$P\ (Correct)$

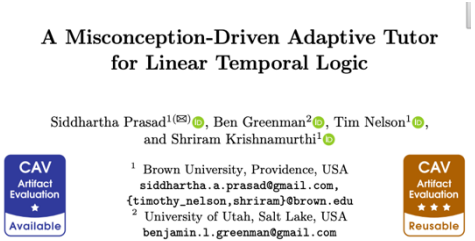
Do learners using the tutor improve?

$$P\,(Correct) \sim (1\,|studentId)\; + [Formula\;difficulty] + \#Qs\;answered$$



The ~~Lazy~~ Amazing Educator

ltl-tutor.xyz



A Misconception-Driven Adaptive Tutor for Linear Temporal Logic

Siddhartha Prasad[1(✉)], Ben Greenman[2], Tim Nelson[1], and Shriram Krishnamurthi[1]

[1] Brown University, Providence, USA
siddhartha.a.prasad@gmail.com, {timothy_nelson,shriram}@brown.edu
[2] University of Utah, Salt Lake, USA
benjamin.l.greenman@gmail.com

CAV Artifact Evaluation ★ Available

CAV Artifact Evaluation ★★★ Reusable

Requires minimal prior instruction

No curricular disruption

Easy to onboard

Available online, also self hostable

Why not just let LLMs write the LTL?

"The red light is on in exactly one state, not necessarily the first."

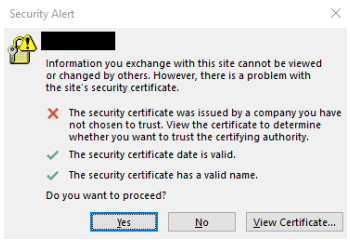
Gemini

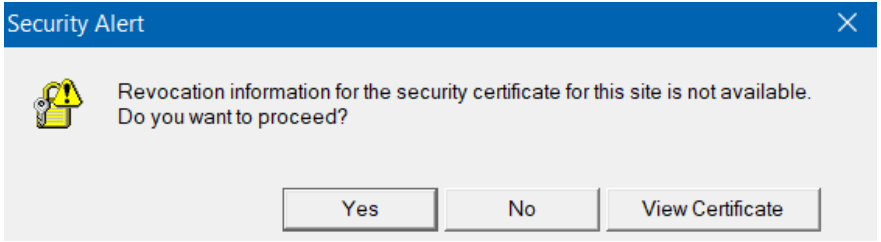
$$(r \ \wedge F(G(\neg r))$$

$$\neg r \, U \, (r \ \wedge \ X(G(\neg r))$$

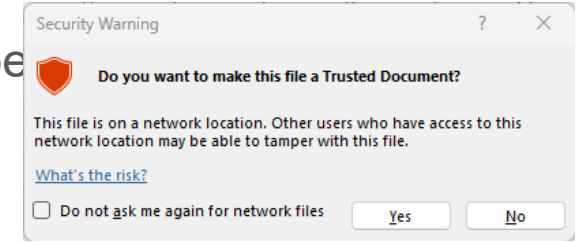
My Position

Humans should be in the loop





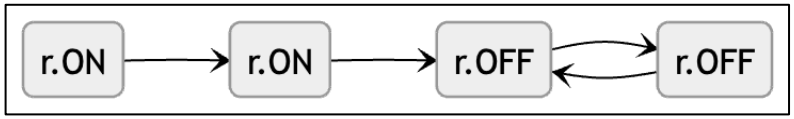
be



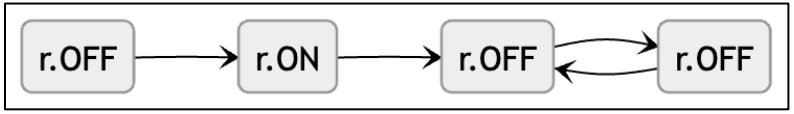
The “**responsible human**“ model

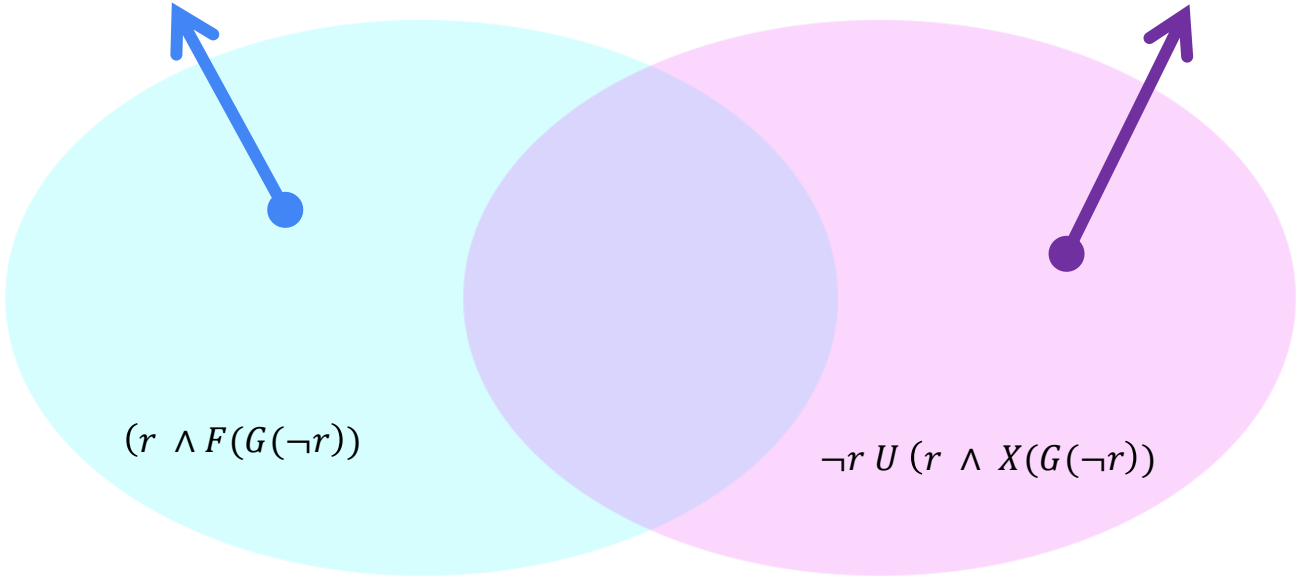
"The Red light is on in exactly one state, not necessarily the first."

Trace 1

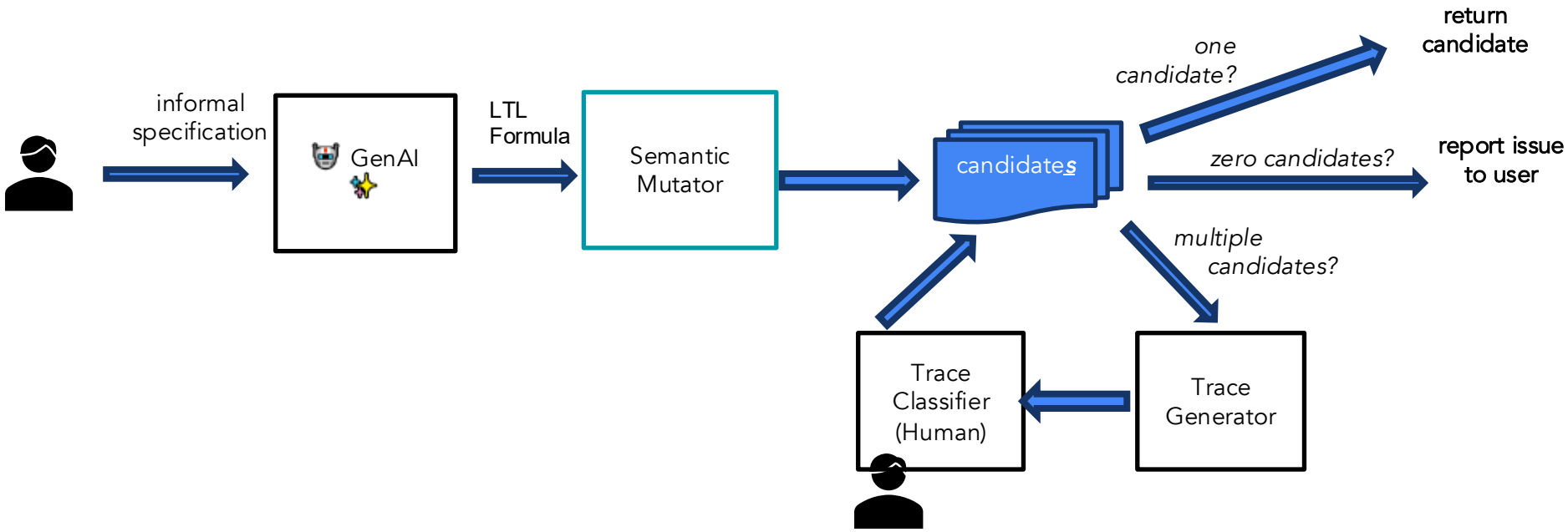


Trace 2





Pairwise Iterative-Choice Knockout



Consumes a non-empty finite set of non-equivalent candidates R and a set of already-classified scenarios $E \subset U$. Produces 2 distinct scenarios.

1. Maintain list T of distinguishing scenarios, initially empty.
2. If $R = \{f_1, f_2, \ldots, f_n\}$ with $n \geq 2$, while $|T| < 2$, for all pairs $\{f_i, f_j\} \subseteq R$:
   a. Sample t from $((\mathcal{L}(f_i) \setminus \mathcal{L}(f_j)) \cup (\mathcal{L}(f_j) \setminus \mathcal{L}(f_i))) \setminus E$.
   b. If t is available, add it to T and E.
3. If $|R| = 1$ or $|T| < 2$, sample f from R.
   a. Sample distinguishing scenarios t^- from $\mathcal{L}(\neg f) \setminus E$ and t^+ from $\mathcal{L}(f) \setminus E$ if available.
   b. If t^+ is available, add it to T and E.
   c. If t^- is available, add it to T and E.
   d. If t^- is unavailable, try to sample scenario t_1^+ from $\mathcal{L}(f) \setminus (E)$. If t_1^+ is available, add it to T and E.
   e. If t^+ is unavailable, try to sample a scenario $t \in \Sigma_f^* \setminus E$. If t is available, add it to T and E.
   f. If $|T| < 2$, then no further distinguishing scenarios can be produced. Sample $2 - |T|$ scenarios from E if available, and add them to T.
4. Return the elements of T.

PICK Informally

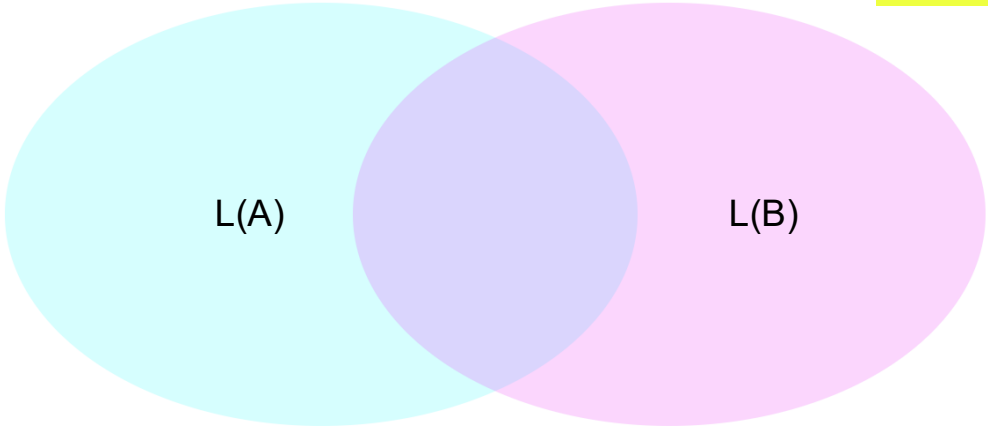
Given a pair of candidate formulae A and B, compute the set differences:

$L(A) \cap \neg L(B)$ and $L(B) \cap \neg L(A)$

Closure properties

Decidable membership

Tractable witness generation



Sample for concrete instances from these distinguishing sets,

ask the human to classify

User Study

RedOnce: Red is on in exactly one state, not necessarily the first.

RedOnOff: The Red light is on for zero or more states, and then turns off and remains off in the future.

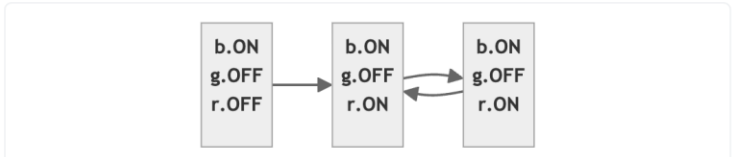
RedBlue: Whenever the Red light is on, the Blue light will be on then or at some point in the future.

All the answers that respondents provided during the misconception identification process

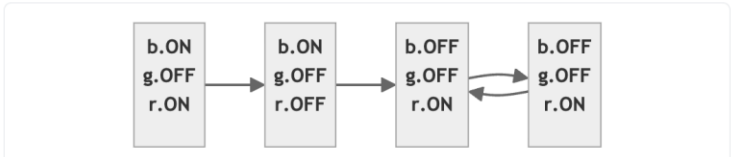
Study Participants: Essentially no knowledge of LTL

There is an instrument panel with three colors: Red, Green, and Blue. **The Red light is on for zero or more states, and then turns off and remains off in the future.** The variable `r` indicates when the red light is on, the variable `g` when the green light is on, and the variable `b` when the blue light is on.

There is an instrument panel with three colors: Red, Green, and Blue. **The Red light is on for zero or more states, and then turns off and remains off in the future.** The variable `r` indicates when the red light is on, the variable `g` when the green light is on, and the variable `b` when the blue light is on.



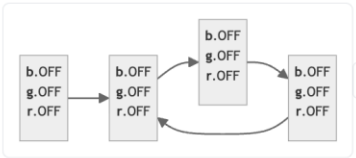
Accept | Reject | Unsure



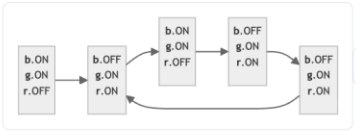
Accept | Reject | Unsure

Your Classifications

Accepted

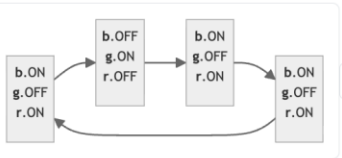


Accept

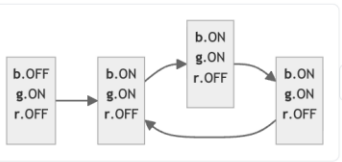


Accept

Rejected



Reject



Reject

Unsure

Little Tricky Logic: Misconceptions in the Understanding of LTL

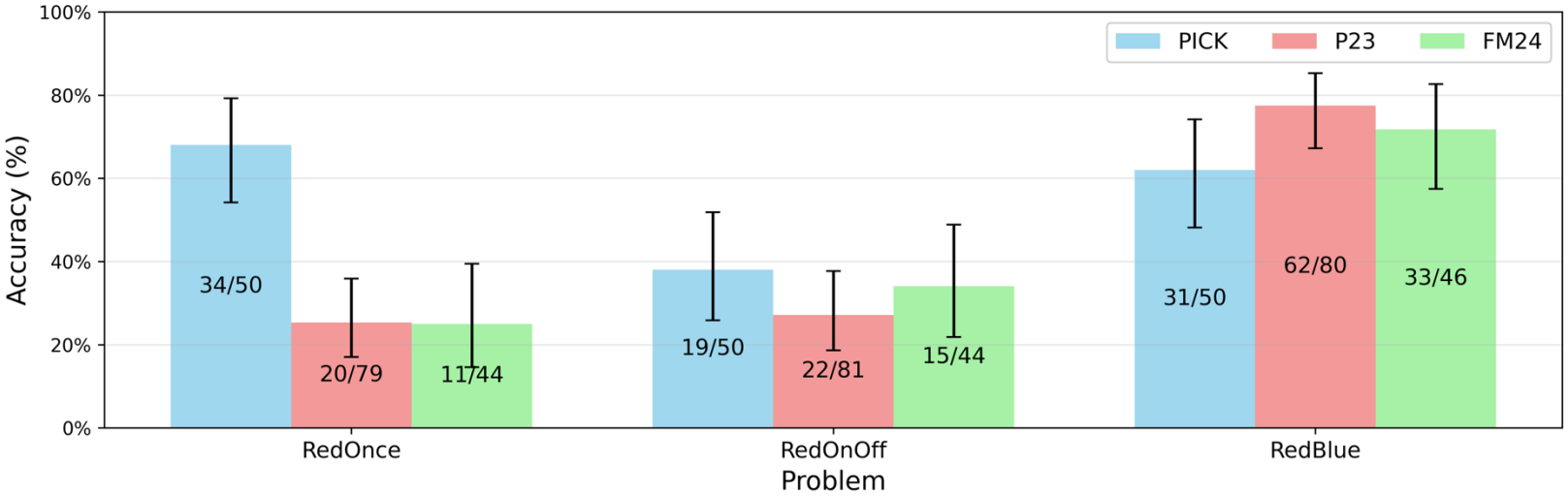
Ben Greenman[a], Sam Saarinen[a], Tim Nelson[a], and Shriram Kr

Misconceptions in Finite-Trace and Infinite-Trace Linear Temporal Logic

FM Artifact Evaluation

FM Artifact Evaluation

Essentially no LTL knowledge, classifying traces only.



RedOnce: Red is on in exactly one state, not necessarily the first.

- ☐ `(!r) U (r & X(G(!r)))` [**Correct**]
- ☐ `F(X(r))`
- ☐ `G(F(r) -> X(G(!r))) & F(r)`
- ☐ `F(r & X(G(!r)))`
- ☐ `G (r -> XG!r)`
- ☐ `F(r) & (r -> X(G(!r)))`
- ☐ `F(r)`
- ☐ `X(F(r) U G(!r))`
- ☐ `(F(r) -> G(!r))`

9 candidates

Median: 13 classifications

RedOnOff: The Red light is on for zero or more states, and then turns off and remains off in the future.

- ☐ `(r U !r) & (G(!r -> G(!r)))` [**Correct**]
- ☐ `r U (!  r) & (!r -> X !r)`
- ☐ `r ->(X(r) | (G !(r)))`
- ☐ `F(G(!r))`
- ☐ `F(r -> X(G(!r)))`
- ☐ `G(r) U G(!r)`
- ☐ `F(r) & G(r -> X(U(!r)))`

7 candidates

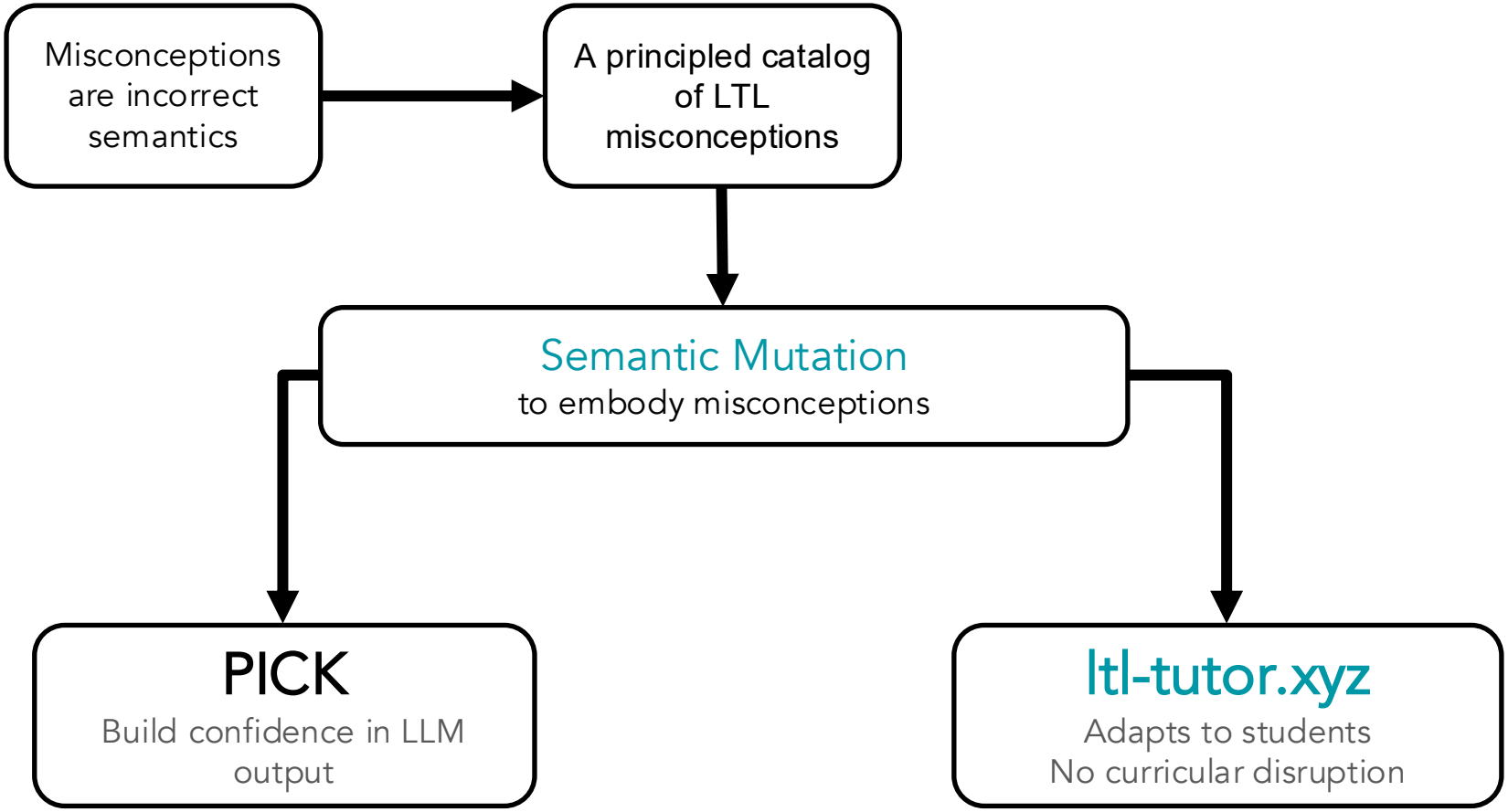
Median: 10 classifications

RedBlue: Whenever the Red light is on, the Blue light will be on then or at some point in the future.

- ☐ `G(X(r) -> (X(b) | F(b)))`
- ☐ `G(r -> F(b))` [**Correct**]
- ☐ `r -> (F(b))`

3 candidates

Median: 8 classifications


Misconceptions are incorrect semantics
A principled catalog of LTL misconceptions
Semantic Mutation
to embody misconceptions
PICK
Build confidence in LLM output
ltl-tutor.xyz
Adapts to students
No curricular disruption